```
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
```

FILEID**BASPOWJJ

```
BBBBBBBB      AAAAAA      SSSSSSSS  PPPPPPPP      OOOOOO    WW      WW          JJ          JJ
BBBBBBBB      AAAAAA      SSSSSSSS  PPPPPPPP      OOOOOO    WW      WW          JJ          JJ
BB      BB   AA    AA   SS        PP      PP   OO      OO  WW      WW          JJ          JJ
BB      BB   AA    AA   SS        PP      PP   OO      OO  WW      WW          JJ          JJ
BB      BB   AA    AA   SS        PP      PP   OO      OO  WW      WW          JJ          JJ
BB      BB   AA    AA   SS        PP      PP   OO      OO  WW      WW          JJ          JJ
BBBBBBBB     AA    AA     SSSSSS  PPPPPPPP     OO      OO  WW      WW          JJ          JJ
BBBBBBBB     AA    AA     SSSSSS  PPPPPPPP     OO      OO  WW      WW          JJ          JJ
BB      BB   AAAAAAAAAA         SS PP           OO      OO  WW  WW  WW  JJ      JJ  JJ      JJ
BB      BB   AAAAAAAAAA         SS PP           OO      OO  WW  WW  WW  JJ      JJ  JJ      JJ
BB      BB   AA    AA           SS PP           OO      OO  WWWW  WWWW  JJ      JJ  JJ      JJ    ....
BB      BB   AA    AA           SS PP           OO      OO  WWWW  WWWW  JJ      JJ  JJ      JJ    ....
BBBBBBBB     AA    AA   SSSSSSSS  PP             OOOOOO    WW      WW    JJJJJJ      JJJJJJ        ....
BBBBBBBB     AA    AA   SSSSSSSS  PP             OOOOOO    WW      WW    JJJJJJ      JJJJJJ        ....
```

```
LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
0000     1          .TITLE  BAS$POWJJ                ; BASIC integer ** integer
0000     2          .IDENT  /1-006/                  ; File: BASPOWJJ.MAR    Edit: LB1006
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY: Basic Support Library
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;
0000    34 ;       This module contains entry points to support exponentiation
0000    35 ;       (** or ^) in BASIC-PLUS-2 for LONG ** LONG.
0000    36 ;
0000    37 ; ENVIRONMENT: User Mode, AST Reentrant
0000    38 ;
0000    39 ;--
0000    40 ; AUTHOR: R. Will        , CREATION DATE: 22-NOV-78
0000    41 ;
0000    42 ; MODIFIED BY:
0000    43 ;
0000    44 ; R. Will,       : VERSION 01
0000    45 ; 1-01  - Original
0000    46 ; 1-02  - Fix comments, change BRW to JMP. RW  7-Dec-78
0000    47 ; 1-003 - Add " " to the PSECT directive.  JBS 22-DEC-78
0000    48 ; 1-004 - Redo the case analysis of the BASE leg 0 case for
0000    49 ;          compatability with the PDP-11.  JBS 24-APR-1979
0000    50 ; 1-005 - Make all external references GENERAL mode addressing.  SBL 15-June-1981
0000    51 ; 1-006 - Check the flags word in the BASIC frame for the setting of
0000    52 ;          the integer overflow bit and use that value as the setting
0000    53 ;          (or clearing) of the PSW.  Also added entry point
0000    54 ;          BAS$$POWJJ_NIV to ensure that the IV bit in the PSL is
0000    55 ;          cleared before giving control to OTS$POWJJ.  LB 15-May-1982
0000    56 ;
```

```
                     58          .SBTTL  DECLARATIONS
                     59 ;
                     60 ; INCLUDE FILES:
                     61 ;
                     62
                     63 ;
                     64 ; EXTERNAL DECLARATIONS:
                     65 ;
                     66          .DSABL  GBL                     ; Prevent undeclared
                     67                                          ; symbols from being
                     68                                          ; automatically global.
                     69
                     70          .EXTRN  OTS$POWJJ               ; OTS$ long ** long exponentation
                     71          .EXTRN  BAS$K_DIVBY_ZER         ; Divide by Zero
                     72          .EXTRN  BAS$$STOP               ; Error reporting routine
                     73          .EXTRN  BAS$HANDLER             ; BASIC condition handler
                     74
                     75 ;
                     76 ; MACROS:
                     77 ;
                     78
                     79 ;
                     80 ; EQUATED SYMBOLS:
                     81 ;
                     82
                     83 ;
                     84 ; OWN STORAGE:
                     85 ;
                     86
                     87 ;
                     88 ; PSECT DECLARATIONS:
                     89 ;
                     90          .PSECT _BAS$CODE PIC, USR, CON, REL, LCL, SHR, -
                     91                      EXE, RD, NOWRT, LONG
                     92
```

```
                     0000     94          .SBTTL  BAS$POWJJ  - BASIC word ** word
                     0000     95 ;++
                     0000     96 ; FUNCTIONAL DESCRIPTION:
                     0000     97 ;
                     0000     98 ;       This routine takes BASE ** EXP, using the following table
                     0000     99 ;       for unusual cases:
                     0000    100 ;
                     0000    101 ;       BASE > 0                        Call OTS$POWJJ, normal case.
                     0000    102 ;       BASE = 0, EXP > 0               Return 0.
                     0000    103 ;       BASE = 0, EXP = 0               Return 1.
                     0000    104 ;       BASE = 0, EXP < 0               Error: divide by zero
                     0000    105 ;       BASE < 0, EXP even              Call OTS$POWJJ with -BASE
                     0000    106 ;       BASE < 0, EXP odd               Call OTS$POWJJ with -BASE, negate result
                     0000    107 ;
                     0000    108 ; CALLING SEQUENCE:
                     0000    109 ;
                     0000    110 ;       CALL result.ww.v = BAS$POWJJ (base.rl.v, exponent.rl.v)
                     0000    111 ;
                     0000    112 ; INPUT PARAMETERS:
                     0000    113 ;
00000004             0000    114         base = 4
00000008             0000    115         exponent = 8
                     0000    116 ;
                     0000    117 ; IMPLICIT INPUTS:
                     0000    118 ;
                     0000    119 ;       NONE
                     0000    120 ;
                     0000    121 ; OUTPUT PARAMETERS:
                     0000    122 ;
                     0000    123 ;       NONE
                     0000    124 ;
                     0000    125 ; IMPLICIT OUTPUTS:
                     0000    126 ;
                     0000    127 ;       NONE
                     0000    128 ;
                     0000    129 ; FUNCTION VALUE:
                     0000    130 ; COMPLETION CODES:
                     0000    131 ;
                     0000    132 ;       long result of exponentiation
                     0000    133 ;
                     0000    134 ; SIDE EFFECTS:
                     0000    135 ;
                     0000    136 ;       Will signal Divide By Zero if its arguments are bad,
                     0000    137 ;       and  OTS$POWJJ may also signal.
                     0000    138 ;
                     0000    139 ;--
                     0000    140
0000'                0000    141 BAS$POWJJ::     .MASK OTS$POWJJ         ; Entry point
                     0002    142                                         ; Since this routine uses no
                     0002    143                                         ; registers and usually transfers
                     0002    144                                         ; control to OTS$POWJJ, we copy
                     0002    145                                         ; its register save mask and then
                     0002    146                                         ; JMP past its save mask and only
                     0002    147                                         ; save the registers once
                     0002    148
                     0002    149 ;+
                     0002    150 ; On a call to BAS$POWJJ, the flags word contained in the BASIC frame
```

```
                                0002   151 ; defines whether integer overflow should or should not be enabled.
                                0002   152 ; The value of the flag should dictate the setting within the PSL.
                                0002   153 ;-
          52    0C AD    D0     0002   154          MOVL    12(FP),R2                ; Fetch the saved frame pointer
51   00000000'GF   DE     0006   155          MOVAL   G^BAS$HANDLER,R1         ; Fetch addr of BAS$HANDLER
          51    62    D1     000D   156          CMPL    0(R2),R1                 ; Check if this is a BASIC frame
                11    12     0010   157          BNEQ    9$                       ; Branch if not a BASIC frame
          52    E6 A2    B0     0012   158          MOVW    -26(R2),R2               ; Fetch flags word from BASIC frame
       52   F7FF 8F    AA     0016   159          BICW    #^XF7FF,R2               ; Clear all but IV bit
                52    B5     001B   160          TSTW    R2                       ; Check if integer overflow is set
                02    13     001D   161          BEQL    8$                       ; Branch if clear
                02    11     001F   162          BRB     9$                       ; Continue as usual
                20    B9     0021   163 8$:      BICPSW  #^X20                    ; Clear integer overflow in PSW
             04 AC    B5     0023   164 9$:      TSTW    base(AP)                 ; Test base relationship to zero
                06    15     0026   165          BLEQ    1$                       ; If base leq 0, do case analysis
       00000002'GF   17     0028   166          JMP     G^OTS$POWJJ+2            ; Transfer control to the OTS$
                                002E   167                                           ; routine to do exponentiation
                                002E   168 ;+
                                002E   169 ; Come here if the base is less than or equal to zero.  We must filter
                                002E   170 ; several special cases, as described above.
                                002E   171 ;-
                23    13     002E   172 1$:      BEQL    4$                       ; Branch if base = 0
             08 AC    DD     0030   173          PUSHL   exponent(AP)             ; Stack EXP as parameter to OTS$POWJJ
          7E 04 AC    CE     0033   174          MNEGL   base(AP), -(SP)          ; Stack -BASE as param to OTS$POWJJ
                52    B5     0037   175          TSTW    R2                       ; Check if IV is set
                09    12     0039   176          BNEQ    7$                       ; Do the regular CALL to OTS$POWJJ
0000006C'GF     02    FB     003B   177          CALLS   #2,G^BAS$$POWJJ_NIV      ; Clear IV before JMPing to OTS$POWJJ
                07    11     0042   178          BRB     10$
00000000'GF     02    FB     0044   179 7$:      CALLS   #2, G^OTS$POWJJ          ; Call integer power routines
          03 08 AC    E9     004B   180 10$:     BLBC    exponent(AP),2$          ; Branch if exponent even
          50    50    CE     004F   181          MNEGL   R0, R0                   ; Exponent odd, negate the result
                04    0052   182 2$:      RET                                      ; and return with it.
                            0053   183 ;+
                            0053   184 ; Come here if the base is equal to zero.  The value we return depends
                            0053   185 ; upon the sign of the exponent.
                            0053   186 ;-
             08 AC    D5     0053   187 4$:      TSTL    exponent(AP)             ; Test the exponent against zero
                09    19     0056   188          BLSS    6$                       ; Branch if exponent lss 0
                03    13     0058   189          BEQL    5$                       ; Branch if exponent is 0
                            005A   190 ;+
                            005A   191 ; Come here if the base is zero and the exponent is greater than zero.
                            005A   192 ; BASIC defines this as 0.
                            005A   193 ;-
                50    D4     005A   194          CLRL    R0                       ; R0 = 0
                04    005C   195          RET                                      ; Return to caller
                            005D   196 ;+
                            005D   197 ; Come here if the base is zero and the exponent is zero.  BASIC defines
                            005D   198 ; this as 1.
                            005D   199 ;-
          50    01    D0     005D   200 5$:      MOVL    #1, R0                   ; R0 = 1
                04    0060   201          RET                                      ; Return to caller.
                            0061   202 ;+
                            0061   203 ; Come here if the base is zero and the exponent is less than zero.
                            0061   204 ; BASIC defines this as an error.
                            0061   205 ;-
       7E   00'8F    9A     0061   206 6$:      MOVZBL  #BAS$K_DIVBY_ZER, -(SP)  ; Divide by zero
00000000'GF     01    FB     0065   207          CALLS   #1, G^BAS$$STOP          ; Report error, never return.
```

```
006C   208 ;
```

```
                     006C   210          .SBTTL  BAS$$POWJJ_NIV
                     006C   211
                     006C   212 ;++
                     006C   213 ; Functional Description:
                     006C   214 ;
                     006C   215 ;       This routine is an internal entry point, called only by BAS$POWJJ
                     006C   216 ;       whose sole purpose in this world is to turn off the integer
                     006C   217 ;       overflow bit in the PSL for the case where the base is less than
                     006C   218 ;       zero, and where the flags bit in the BASIC frame indicate to turn
                     006C   219 ;       off integer overflow.
                     006C   220 ;
                     006C   221 ; Calling Sequence:
                     006C   222 ;
                     006C   223 ;       CALL BAS$POWJJ (base.rw.v, exponent.rw.v)
                     006C   224 ;
                     006C   225 ; Input Parameters:
                     006C   226 ;
                     006C   227 ;       None
                     006C   228 ;
                     006C   229 ; Output Parameters:
                     006C   230 ;
                     006C   231 ;       None
                     006C   232 ;
                     006C   233 ; Side Effects:
                     006C   234 ;
                     006C   235 ;       OTS$POWJJ may signal
                     006C   236 ;--
                     006C   237
                     006C   238 BAS$$POWJJ_NIV::
             0000'   006C   239                   .MASK   OTS$POWJJ       ; Entry point
          20   B9   006E   240                   BICPSW  #^X20           ; Clear IV bit in PSL
00000002'GF   17   0070   241                   JMP     G^OTS$POWJJ+2   ; Transfer control to the OTS$
                     0076   242                                           ; routine to do exponentiation
                     0076   243
                     0076   244          .END
```

```
BAS$$POWJJ_NIV   0000006C RG    01
BAS$$STOP        ********   X   00
BAS$HANDLER      ********   X   00
BAS$K_DIVBY_ZER  ********   X   00
BAS$POWJJ        00000000 RG    01
BASE           = 00000004
EXPONENT       = 00000008
OTS$POWJJ        ********   X   00
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_BAS$CODE	00000076 (118.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.10	00:00:00.46
Command processing	111	00:00:00.49	00:00:02.92
Pass 1	69	00:00:00.62	00:00:01.69
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	56	00:00:00.45	00:00:00.90
Symbol table output	2	00:00:00.02	00:00:00.02
Psect synopsis output	2	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	271	00:00:01.69	00:00:06.01

```
The working set limit was 900 pages.
2800 bytes (6 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 8 non-local and 9 local symbols.
244 source lines were read in Pass 1, producing 8 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	0

```
0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:BASPOWJJ/OBJ=OBJ$:BASPOWJJ MSRC$:BASPOWJJ/UPDATE=(ENH$:BASPOWJJ)
```

BASOPEN
LIS

BASPOWJJ
LIS

BASPOS
LIS

BASPOWDJ
LIS

BASOPENDE
LIS

BASPOWGG
LIS

BASPOWHH
LIS

BASPOWRJ
LIS

BASPOWII
LIS

BASPURIOB
LIS

BASPOWDD
LIS

BASOPENZE
LIS

BASPOWDR
LIS

BASPOWGJ
LIS

BASPOWRD
LIS

BASPOWHJ
LIS

BASPOWRR
LIS